DARPA Initiative in Concurrent Engineering (DICE)

Phase 5

Collaboration Technologies for Concurrent Engineering

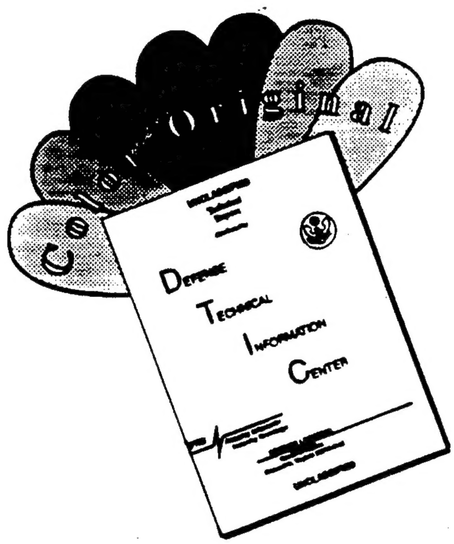
Final Report

2 MAY 96

Sponsored by

Defense Advanced Research Projects Agency
Defense Sciences Office
Contract #MDA972-89-C-0020

DISCLAIMER NOTICE



THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF COLOR PAGES WHICH DO NOT REPRODUCE LEGIBLY ON BLACK AND WHITE MICROFICHE.

TABLE OF CONTENTS

List of Figures

List of Tables

EXECUTIVE SUMMARY

Large systems have become far too complex and multidisciplinary to be understood in detail, much less developed by individuals. Recognition of the importance of life-cycle engineering has multiplied this complexity.

The complex problems arising in system lifecycle design and development must be addressed by formal and informal teams, which introduces the issue of productive and harmonious collaboration. MECE was developed to to meet this need.

Any list of attributes of environments which foster collaboration starts with shared motivation and goals. MECE can help in reaching these agreements which are essential to success. The users must want and need to work together.

The essential attributes of collaboration supported by MECE are

1) sharing motivation and goals.

2) expressive means to develop, express, and display ideas and issues, and comment on those of others

3) easy expressive communications over distance and time

4) shared information/models relating to the task at hand

5) shared understanding of this information

6) accessible shared "institutional memory"

Table 1.1 indicates how these attributes of successful collaboration are realized in MECE.

ATTRIBUTES	MECE REALIZATION
1) Sharing motivation and goals	Shared information about design intent and rationale
2) means to develop, express, and display ideas and comment on issues	Agents for multimedia information generation and a shared workspace to display it for integration and comment
3) Easy expressive communications over distance and time	Appropriate channels and bandwidth for "live" and stored communication, provided by conferencing, shared whiteboards, multimedia e-mail, databases
4) Shared information/ models relating to the task at hand	Shared displays and product views, shared analyses and commentary, indexed, linked information storage, engineering notebooks
5) Shared understanding of task information	Open forum for discussion, shared contexts and vocabularies
6) Accessible shared "institutional memory"	Indexed, linked information storage as part of the project notebook

TABLE 1.1 - FACTORS IN SUCCESSFUL COLLABORATION

MECE is specialized in that it is an <u>engineering</u> collaboration environment, which means that its capabilities are extended over those of most collaboration techniques to share a wider variety of representations and deal with notification and version control.

Table 1.2 provides a compact description of MECE capabilities as they correspond to the collaboration attributes of Table 1.1, and the agencies (agents and tools) which underlie them.

MECE CAPABILITIES	ATTRI-BUTES	MECE AGENCIES
Supports individual work and documentation at arbitrary levels of formality	1), 2), 4), 5)	Authoring and Navigation agents, private notebooks
Supports group work (information and activity sharing) and documentation at arbitrary levels of specialization and formality	all	Authoring and Navigation agents, shared informal and formal notebooks, document capture and markup, multi-media e-mail, electronic conferencing
Notifies interested parties of new entries and changes	1) , 2), 3), 4)	Authoring agent
Supports collaboration over distance and time	2), 3)	teleconferencing, multimedia e-mail, shared whiteboards
Maintains a permanent record of design/ engineering activities, decisions,and rationale with version control	5)	engineers' notebooks, capture of text/graphical documents, object database
Provides for indexing, filtering, and retrieval of database entries in large notebooks	5)	Authoring and Navigator agents for viewing, indexing, searching, and traversing graphical hypertext node/directed-link displays
Prepares and launches applications using tools such as NetBuilder and SimBuilder		"hot links" to tools from notebook entries
Supports evolutionary - development and integration of new facilities		"open" environment provided by agent oriented architecture

Table 1.2 - Attributes and Agencies in Engineering Collaboration

MECE is an *environment* provided by *agents*, which use *tools*. This architecture makes it easy to enhance existing agents and add new ones without revising the overall environment.

AGENTS, TOOLS, and IMPLEMENTATIONS

AGENTS	TOOLS	IMPLEMENTATION
Authoring, indexing, notification	hypertext, graphics, multimedia page capture, keyword indexes and filters, multimedia e-mail	ABYSML, GIF, MIME, MPEG
Navigation	hypertext links, keyword searches and filters	ABYSML
Communications	teleconferencing, multimedia e-mail, shared whiteboards	MIME, MPEG, voice
Database storage & retrieval (with version control)	DIRCCE	RCS
Web interface	Netscape	HTML

Table 1.3 - Agents, tools, and implementations

There are other "collaboration environments" on the market. By MECE standards, they are all incomplete. Table 1.4 compares two other typical systems with MECE. Note that VNS (Virtual Notebook System) is an engineering environment and was a predecesssor to MECE. Lotus Notes is a very general system using compound documents to facilitate the communications aspect of collaboration. It is an enhanced form of electronic mail.

Capability	VNS	Lotus Notes	MECE
Support to group work (information and activity sharing)	Notebooks, word processors, hypertext, drawing tools	"Notes", uses PC, Macintosh text and graphics tools	Notebooks, hypertext Authoring and Navigation environments
Communications	multimedia e-mail	multimedia e-mail	Teleconferencing, multimedia e-mail, shared whiteboards
Accessible shared "institutional memory"	Notebooks	Database of forms	Notebooks indexed by keywords and hypertext links
Prepare and launch external applications	yes	N/A	Hot links from notebooks
Permanent record of design/ engineering activities and decisions	Notebooks, database	Database	engineers' notebooks, object database
notification of additions and changes	not inherent	N/A	Authoring agent
version control	not inherent	N/A	DIRCCE
Development and integration of new facilities	fairly easy	N/A	Open environment for new agents

Table 1.4 - MECE compared with other useful collaboration environments

1.0 INTRODUCTION

DICE 5 was a major step in the long history of Lockheed Missiles and Space and Lockheed Skunk Works work in collaboration and concurrent engineering. Lockheed Independent Research and Development, and DARPA contracts in these areas are described below, together with some of the fruits of this work. The major outcome of the Lockheed DICE 5 contract was MECE, the Multimedia Engineering Collaboration Environment, which in many ways exploited the previous results. It will also be a platform for future enhancements to the collaboration process.

1.1 Background: Lockheed and DICE

In 1992 the R&D division of Lockheed Missiles and Space was awarded a contract to field DICE software to a selected pilot organization. Prior to this DICE 4 contract the Lockheed Artificial Intelligence Center had been researching CE and design topics through DARPA contracts and Independent Research and Development (IRAD) programs.

• *AI-Based Computational Framework for Concurrent Engineering:* an independent development project to apply object-oriented database modelling techniques to integration of design decision systems.

• *Knowledge-Centered Design:* research on capturing expertise as systems are designed and built; automatically acquiring a representation of knowledge during the design; providing tools that allow users to easily express designs and design decisions; and supporting the automatic generation of explanations.

• *Information Frameworks and Architecture:* joint work between Enterprise Integration Technologies (EIT) and the Lockheed AI Center to support a unique computer-based engineering environment for sharing information among members of a large geographically distributed team of engineers working on a large DOD project.

• *NVisage:* an object-oriented software architecture for digital circuit design and simulation. Nvisage was one of several technologies demonstrated to DARPA on the Palo Alto Collaborative Testbed (PACT).

• *Shared Knowledge-Based Technology for Re-Engineering:* a DARPA funded effort to develop reasoning techniques and ontologies for incremental refinement of design descriptions.

• *Shared Design and Engineering Knowledge* (SHADE): a DARPA funded effort to facilitate the exchange of engineering results and knowledge via an infrastructure of communications and shared representations, using and extending KIF (Knowledge Interchange Format) and KQML, Knowledge Query and Manipulation Language).

There are also several efforts in Lockheed product divisions in areas related to CE. In particular, *Computer Integrated Engineering and Manufacturing* (CIEM) is a major Lockheed initiative to automate and integrate the design, analysis, and manufacturing elements of mechanical product development in a concurrent engineering environment. This system is under development by the Space Systems Division(SSD) to reduce product development time and cost while simultaneously improving product performance and quality.

1.1.1 Engineering Collaboration at CDR

The Stanford Center for Design Research First and Next-Cut systems address the problems in automated support to collaboration in concurrent engineering. This background contributed to the development of the First-Link Cable Knowledge system for automated cable routing in highly constrained environments. This uses 3-D CAD models to represent the volumes to be traversed, and an agent oriented approach to routing the cables.

1.1.2 Electronic Roundtable

Starting in 1989, the Lockheed Skunk Works began research to improve communications in product development organizations by using electronic means to improve the already excellent communications which occur within the Skunk Works organization by virtue of physical co-location and a unique organizational culture. This work included the development of an integrated electronic e-mail and shared notebook environment which included simple graphics capability, the use of hyper-text links with embedded information about the type of connection implied by the link, and simple filtering capabilities for navigation. This was used on a small pilot project, and provided considerable information about its basic feasibility and utility.

In 1991, additional work was done using a commercial desktop publishing tool to provide active day to day working communication on a more substantial pilot project involving intensive integrated product development for the conceptual design of a hypothetical supersonic business transport aircraft. This activity lasted 6 months and was unusual in that it while it was a conceptual design effort, specific parts of the aircraft which were considered to provide substantial opportunity for cost saving were designed to a relatively high level of detail. This work demonstrated the advantages, even for a small co-located design team, of a written, persistent, and relocatable forum for design idea generation and decision making. The more rigorous communication provided by this mechanism resulted in not only better decisions and designs, but, somewhat surprisingly, improvement in creativity and innovation. This was because capturing and distributing design rationale exposed previously hidden and invalid assumptions. Eliminating these provided more freedom to the team and resulted in more innovative designs. The knowledge gained in this pilot project along with the aforementioned prototype electronic notebook system helped provide a foundation for the definition of the MECE system.

1.1.3 DICE - Phase 4

The purpose of the Lockheed DICE Phase 4 program was to evaluate, enhance, harden, and integrate existing concurrent engineering (CE) tools. This pilot study was part of a five phase DICE program to research, develop, implement, test, and transition CE tools and technologies to industry.

Lockheed was funded by DARPA to support the Composites Automation Consortium (CAC) in the use of DICE software and technologies to develop a composite parts fabrication system. The CAC was a consortium of small, medium, and large commercial companies, research laboratories, and university affiliates which had not previously done machine design as a group. The CAC was geographically distributed, with several members performing the machine design concurrently at separate sites.

CAC chose to be the pilot organization for the Lockheed contract as part of their Composites Automation Manufacturing Initiative (CAMI) program, which was tasked

with developing an automated machine system that would enable more cost-effective manufacture of key, representative, advanced composite structures.

Selection of the CAC provided an opportunity to apply CE technology for use on a practical and real problem, and for wide dissemination of the results in industry. Working with the CAC, the Lockheed DICE team evolved a CE system that was used by their geographically dispersed members.

During Phase 4 of DICE, the CAC was in the process of conceptual design of an innovative composite manufacturing system and used the CE system developed by Lockheed for that purpose. The benefit to the DICE program was the continuous evaluation of the effectiveness and acceptability of the tools and framework provided.

The Lockheed team felt that placing emphasis on conceptual design was important since it is the most critical period of the CE process. The most far-reaching design decisions are made during concept formation, and the principal interactions between disciplines and perspectives are uncovered and considered during concept evaluation. Furthermore, CE tools developed to support conceptual design will also be applicable to all other design phases.

Lockheed began the pilot task by surveying each of the CAC/CAMI sites to determine their network, software, and hardware capabilities. BDM, a Lockheed sub-contractor, then executed a comprehensive Quality Function Deployment (QFD) survey to determine CAC concurrent engineering needs. Lockheed, BDM, Draper Laboratories, and the CAC then identified those CE tools and applications that were applicable to CAMI.

The primary CE tools provided to and used by the CAC were daily teleconferencing, Distributed Revision Control for Concurrent Engineering (DIRCCE), and the Virtual Notebook System (VNS). These tools were selected from the pool of DICE CE software, Lockheed CE software, and available commercial CE software on the basis of how well they matched the CAC/CAMI needs determined in the QFD survey. During the course of the pilot study, incrementally enhanced versions of each tool were fielded and were subjected to user critique.

The CE system became an integral part of the day to day operations of the CAC/CAMI team. Some problems existed with each of the specific tools, but the CE system was composed of complementary tools, fulfilling many of the CAC's core needs. Since this was a new venture, Lockheed was unable to measure the previous "as is" design process for CAMI to determine the increase in productivity resulting from the use of the new CE tools. Lockheed was, however, able to view the user acceptance of the system as part of day to day operations. This was a tribute to the acceptance and usefulness of the CE system fielded by the Lockheed-led DICE team.

1.2 CE and the Need for Distributed Collaboration

There are several reasons why electronic collaboration is essential to concurrent engineering. First, since concurrent engineering is necessarily a team activity, collaboration is part of the process. Next we ask what are the requirements on collaboration in an engineering context:

1) means to develop, express, and display ideas, issues, positions on issues, propose a design, etc. in "public" ways that support comment, help, and criticism from colleagues

2) means to comment on and reshape the proposals of others

3) means to interact directly and remotely to express and resolve different viewpoints

4) means to keep a large amount of current relevant information "at hand" to be consulted and coordinated to arrive at decisions and further views

5) means to keep all project information available in permanent storage under version control, for reference on demand

Several difficulties currently hinder the fulfillment of these basic requirements. In the case of 1) and 2), development and display of engineering information, shared visual information spaces are indispensable. Limitations on and segregation of different types of media and representations limit the expressiveness of a designer or other engineer. Designers are often limited to text and sketching as their primary means of communicating design changes, comments, etc. As a rule hyperlinks are hard, if not impossible, to create and offer a limited view of the information structure. A tight and inflexible integration of tools limits the designer's tool set to those they are accustomed to using.

The issue addressed in 3) is communication over space and time, which must be easy and expressive (e. g. teleconferencing or multimedia mail) to meet the needs of engineering collaboration. Geographic separation must be overcome to share information in a timely fashion and to jointly discuss and mark up documents and designs.

The problem of context addressed in 4), the simultaneous accessibility of all the current and relevant information, is best managed by hypertext retrieval, backed-up by more conventional search and presentation techniques. At present, once information is filed, it often can not easily be located, even by the author.

All four of the above collaboration issues involve information sharing in an environment where product design and development teams are becoming increasingly larger and more geographically distributed.

1.3 MECE Goals and Solutions

Electronic collaboration is essential to concurrent engineering, and there are a number of capabilities that are essential to effective electronic collaboration. The goal of MECE is to provide the capabilities needed to meet the requirements outlined in the previous section. To that end, MECE seeks to facilitate the collaborations that occur among a geographically-distributed product design team and to provide a shared, collective project memory through which designers and other engineers may capture and share product information.

This section is organized using the general requirements stated in Section 1.2:

Requirement 1) - means to develop, express, and display ideas, issues, positions on issues, propose a design, etc. in "public" and immediate ways to support comment, help, and criticism from colleagues, and

Requirement 2) - means to comment on and reshape the proposals of others

• to alleviate the problems associated with developing, capturing, and display of design information, MECE offers the following solutions. First, MECE alleviates the problem of media segregation by utilizing high bandwidth communication and by integrating other

media in addition to text and simple drawing primitives. Additional media types include images, audio, video, and active links to external applications. Furthermore, MECE provides an easy-to-use link creation mechanism based on the familiar cut and paste paradigm. This enables multiple views of the structure of the information. Finally, MECE allows designers to easily integrate information produced by design tools, rather than requiring the integration of the tools. This flexibility is achieved either by utilizing the snapshot capability to include a static image of data or by including an active link to a dynamic model of the data.

Requirement 3) - means to interact directly and by "mail" to express and resolve different viewpoints

• MECE provides a unified environment for collaborating over both space and time regardless of whether the design team is collocated or geographically separated. It is based on high-bandwidth teleconferencing, shared whiteboards, and multi-media e-mail for rapid exchange of engineering information, including video, color, animation, and sound as well as text and 2- and 3-D representations

Requirement 4) - means to keep a large amount of current relevant information "at hand" to be consulted and coordinated to arrive at decisions and further views

• MECE presents the contents of notebooks as filtered lists of topics and webs of filtered relevant entries connected by directed hypertext links. To assist designers in staying current, MECE provides an automatic notification mechanism to inform interested parties whenever new information is published to the notebook.

Requirement 5) - means to keep all project information available in permanent storage under version control, for reference on demand

• In order to assist designers in locating information, MECE provides a rich set of indexing and search tools over a shared database.

2.0 The MECE Approach to Promoting Collaboration

The long range goal of the DARPA Initiative in Concurrent Engineering (DICE) is to enable industries to produce better products at less cost in a shorter period of time. A technology that achieves this goal should enable all the disciplines important in the life cycle of a product or system to cooperate in the definition, planning, design, manufacture, maintenance, refinement, and disposal of that product. The first several phases of DICE involved analyzing CE, the design process, and the interactions that occur during the product life cycle. Using this information, several CE tools were then written to assist in the design process. Some of these tools were then fielded in real world design situations for testing. These pilot efforts confirmed that CE did improve the quality of the example products at a lower cost in less time.

Lockheed's DICE Phase 4 task focused on the evaluation, hardening, and integration of CE tools for the development of composite parts fabrication systems, and on the metrics for measuring the success of the enhanced system. Phase 5 focused on collaboration technologies for CE. Lockheed's contribution to solving the problem of improving support to collaborative engineering is the MECE environment, described in detail in section 2.1.

MECE is based on three major thrusts in DICE: sharing product information, overcoming problems resulting from geographic dispersion of teams; and generating, incorporating, and propagating lifecycle constraints during design.

Sharing information enables all the essential collaborators to participate in decisions, and others to be informed when they are made. It is particularly important in conceptual and preliminary design. It is often asserted that 80% of a product's life cycle costs are determined during the first 20% of product development. Information sharing enables high quality early decisions, arrived at only after they have been considered from all the relevant viewpoints. Through collaboration/ consultation the Implications of these decisions can be recognized before they become part of the design, avoiding most of the causes for redesign and permitting later changes at a lower cost.

One of the focuses of MECE has been on linking geographically dispersed teams. This is required because there is an increasing trend for collaboration between multiple companies in the design and manufacture of products. This is also an issue within a single large corporation where tasks are divided between different divisions.

MECE promotes the incorporation of life cycle considerations into the design and documentation of the product. This is supported by the documentation of design requirements, notification when decisions are made or constraints change, and a shared data model that captures complete descriptions of the product throughout its lifetime.

2.1 The MECE System

Design teams are becoming increasingly dispersed, emphasizing the need for technologies that enable them to collaborate both asynchronously and in real-time. In response to this need, Lockheed developed the Multimedia Engineering Collaborative Environment (MECE). MECE meets this need by facilitating informal, multimedia information interchanges between teams of geographically-distributed designers and other engineers, and by providing a shared, collective memory consisting of not only all the design rationale for a project but also all the individual postings, and the informal

discussions between two or more of the team members as they occur. This shared repository of information is referred to as an electronic *project notebook.*

2.1.1 Applying MECE - When and How

MECE is an informal collaboration system to be used as freely as one currently uses electronic mail. MECE is used in any situation where a geographically-distributed group of people have a need to share information, especially in cases where multimedia information is to be shared. Engineering has usually been the example, but, many other such applications exist. The health professions, law enforcement, and assembly of a national newspaper represent a few of many possibilities.

MECE is much more than a single tool. Rather it is an integrated environment in which the engineer conducts most of his day-to-day business. MECE is an open environment in that it is not designed to replace the tools that the engineer currently uses (e.g. CAD, Excel, etc.) but to replace the environment in which engineers document their work, exchange ideas and product data, consult others, and exchange multimedia information. MECE makes the process of incorporating externally generated product data quick and easy.

2.1.2 User Interaction With MECE

The MECE desktop environment consists of a top level interface from which all the other components are accessible **[Figure 1]**. MECE provides facilities for generating, capturing, displaying, annotating, and storing information, and for retrieving information from a project notebook. To perform information generation, capture, and retrieval, MECE provides agents for multimedia authoring, publication, and navigation.

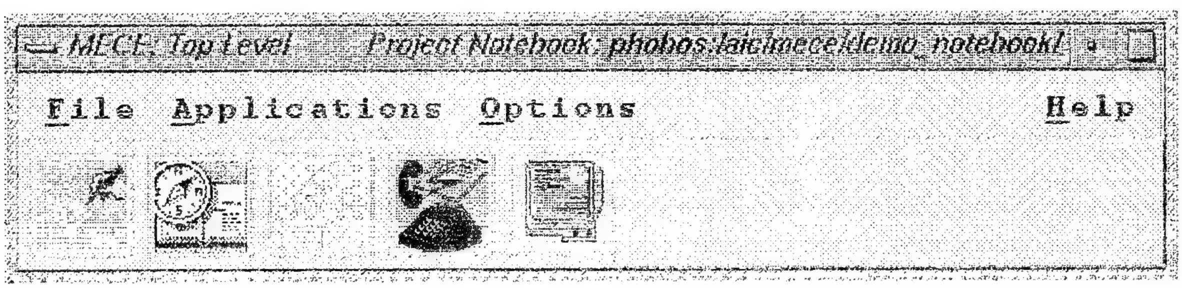


Figure 2.1 - The MECE top level window provides access to the project notebook

Information generation and capture is achieved with the MECE authoring tool **[Figure 2]**. The authoring tool provides a text and graphics editor for generation of new entries or annotations of "captured" material. Any data generated using desktop tools external to MECE can be captured via "snapshot". Examples of external tools are CAD tools, spreadsheets, and word processors. Once the information has been imported into the authoring environment, the engineer uses the text annotation and graphics functions to "mark up" the snapshot. Audio and video clips may also be attached to the entry. In addition to a static image of computed results, the engineer can add an active link to a dynamic model by associating the corresponding data set with the external tool that generated the results. Clicking on the active link allows the user to view and manipulate the dynamic model using this tool. The engineer can also create hyperlinks to existing multimedia entries in the project notebook.

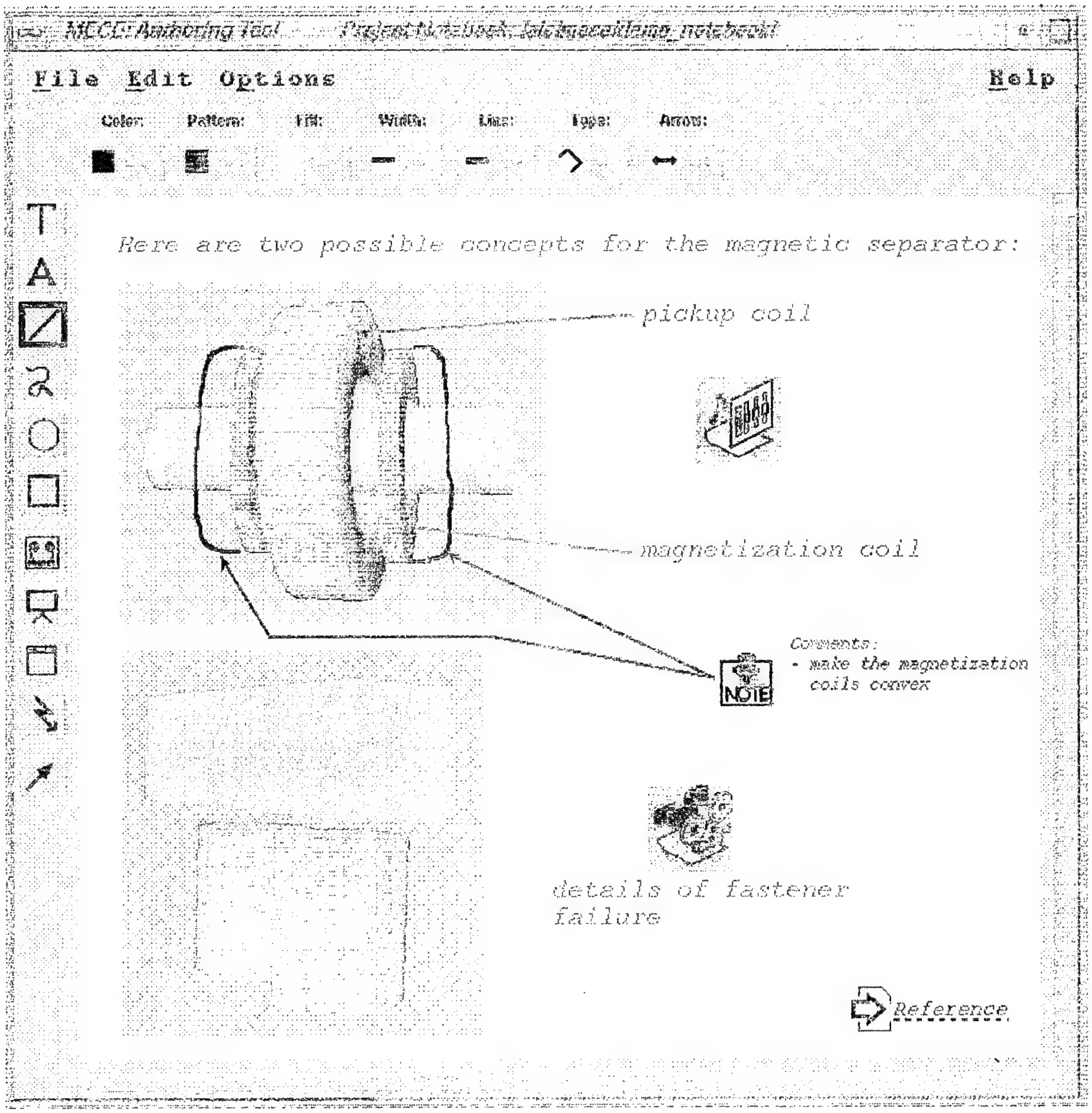


Figure 2.2. The MECE authoring tool provides information capture

After information has been generated or captured and marked up, the publication mechanism stores the new entry in the project notebook and automatically notifies interested colleagues that a new entry has been created [**Figure 3**]. Simultaneously, the entries are indexed in the project notebook based on header information provided by the author. Notification can occur in two ways. First, an author of an entry may explicitly specify the people who should receive a notification, at publication time. Second, users can create one or more interest profiles that specify the types of entries for which they are interested in receiving a notification. Then at publication time, the set of interest profiles for a project notebook is scanned for any that match the header information for the new

entry. Finally each owner of a matching profile receives a notification in the form of an electronic mail message using the Multipurpose Internet Mail Extensions (MIME) format. The receiver of the MIME mail message can then browse the MECE entry, viewing all its multimedia elements.

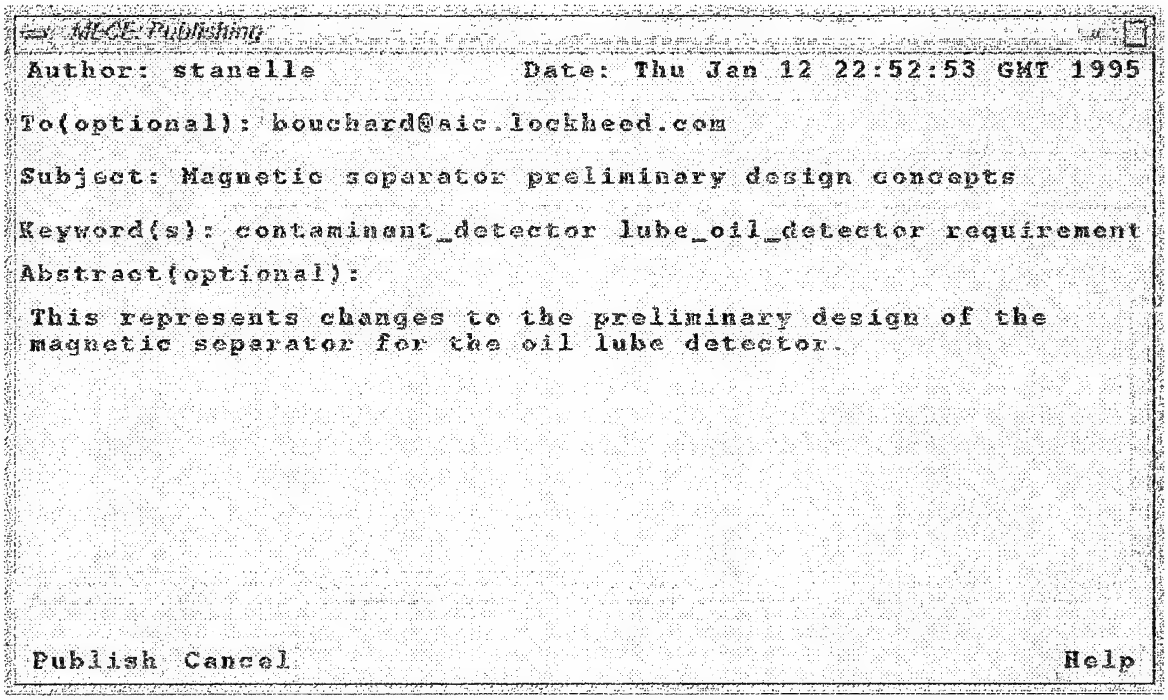


Figure 2.3. The MECE publication mechanism indexes a new entry in the notebook

Information retrieval is performed with the MECE navigator **[Figure 4]** agent. The navigator provides a powerful search engine for searching the project notebook for entries of interest. Queries are constructed by completing a form which specifies a number of search parameters **[Figure 5]**. Search control parameters include author, keywords, type, subject, and a range of from-to dates. In addition, the user can search on available link information. For example, a query may be issued for all entries in the notebook that *depend on* the specified entry.

Once the query is initiated, MECE searches the project notebook and returns all those entries that satisfy the search criteria. Results are presented both graphically and as a list. The graphical presentation consists of a set of nodes and directed arcs. Each node represents an entry, and the text within the node corresponds to the entry subject or title that was supplied by the user at publication time. Each directed arc represents a hyperlink between two entries, with the direction of the arrow indicating the target of the link. The list form of the search results consists of header information corresponding to entries in the graph.

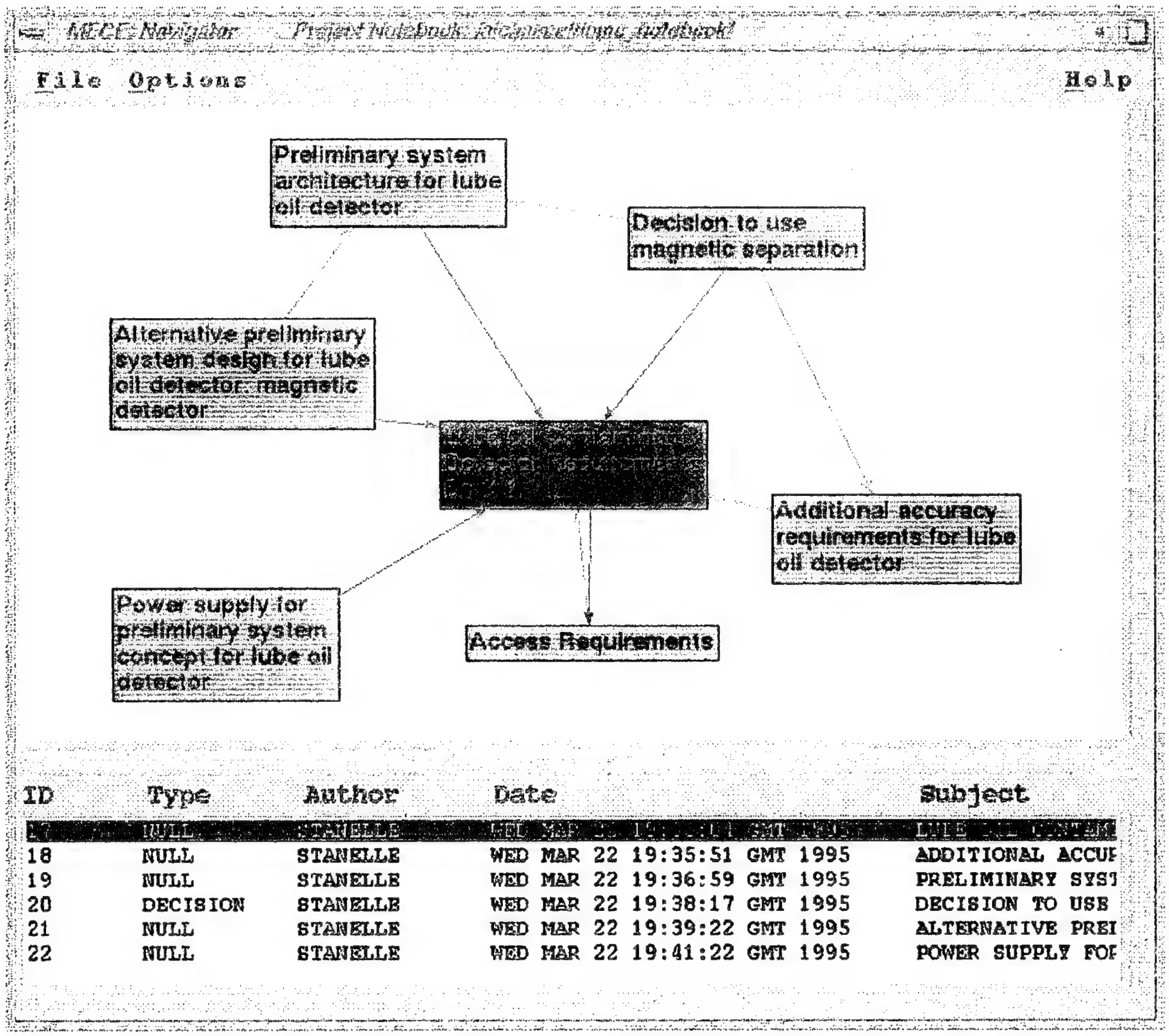


Figure 2.4. The MECE navigator supports information retrieval

To view an entry in the project notebook, the user can either click on an entry in the graph or double click on an entry in the list. MECE will display the selected entry in a viewer [**Figure 6**]. Hyperlinks to other entries can be activated from the viewer version. Link targets may then be displayed in the current viewer, or a new viewer may be opened to display the link target. A history capability allows the user to return to previously-viewed entries without repeating the search.

Another important component of MECE is the taxonomy manager. This is a general tool for managing and maintaining various hierarchies within the MECE system. Examples of entries include project keywords, team members, and product structure. Both keyword and personnel manager hierarchies have been implemented as specific instances of the taxonomy manager.

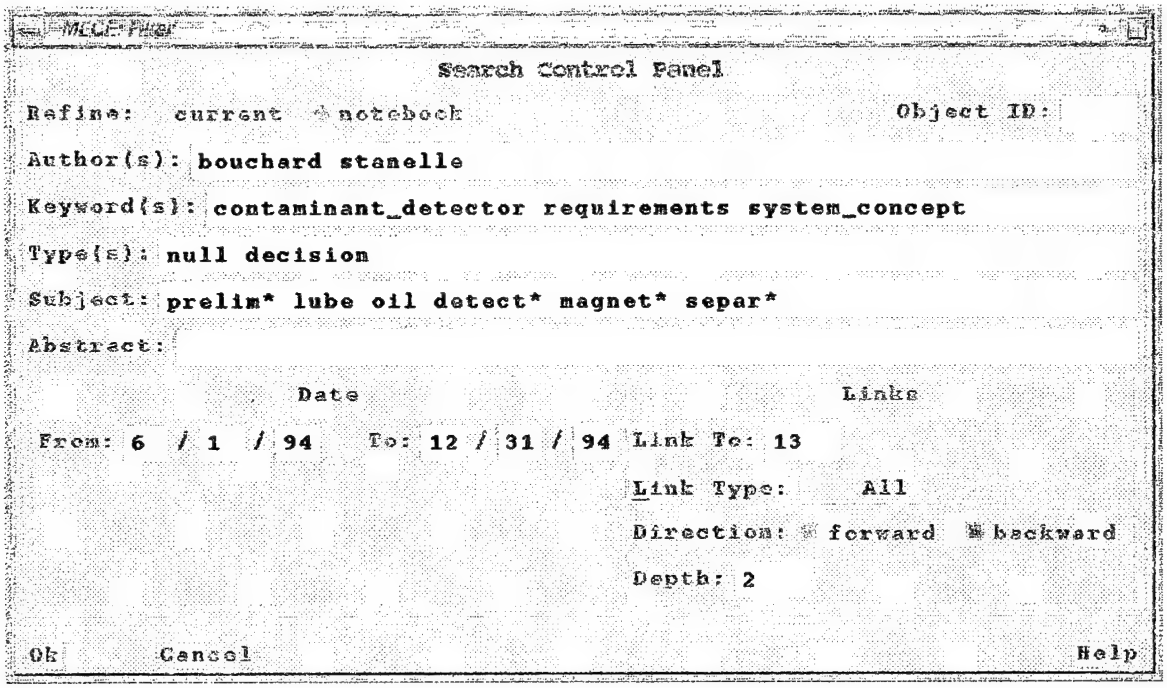


Figure 2.5 - Search Queries are generated by specifying parameters

The primary function of the keyword manager is to maintain an official list of keywords for a project [**Figure 7**]. The reason for maintaining an official keyword list is to avoid ambiguity and redundancy in notebook searches and entry publication. These are possible because, whenever a new entry is published or whenever a search is conducted, the user has the opportunity to specify keywords. Rather than allowing users to use arbitrary keywords that have the same or similar meaning, MECE users are restricted to choosing keywords from the keyword manager list. The keyword manager also has a facility for adding new keywords and synonyms for defined keywords. Built-in checking eliminates the re-definition of a keyword that is already defined or that is the synonym of another keyword. Similar checking occurs for the intentional definition of a synonym for a keyword. As a project progresses and as more keywords are defined, their classification as more general and more specific within a given category leads to the creation of hierarchies.

The personnel manager hierarchy is similar to that of the keyword manager, except that it maintains an official list of project team members showing who reports to which team member.

2.1.3 Implementation

MECE has so far been implemented on SUN, SGI, and PC platforms. On the UNIX platforms (SUN, SGI), the MECE interface is written in Motif 1.2, and the backend code is written in C++. On the PC platform, the Microsoft Foundation Class (MFC) libraries are used to implement the interface, and Visual C++ is used to implement the backend code. Since MECE is written using standard development packages (Motif/C++) on SUN and SGI platforms, we expect that it can be readily compiled on other UNIX platforms (e.g. IBM RS6000, HP, etc.). Because of lack of time and resources, this has not yet been attempted

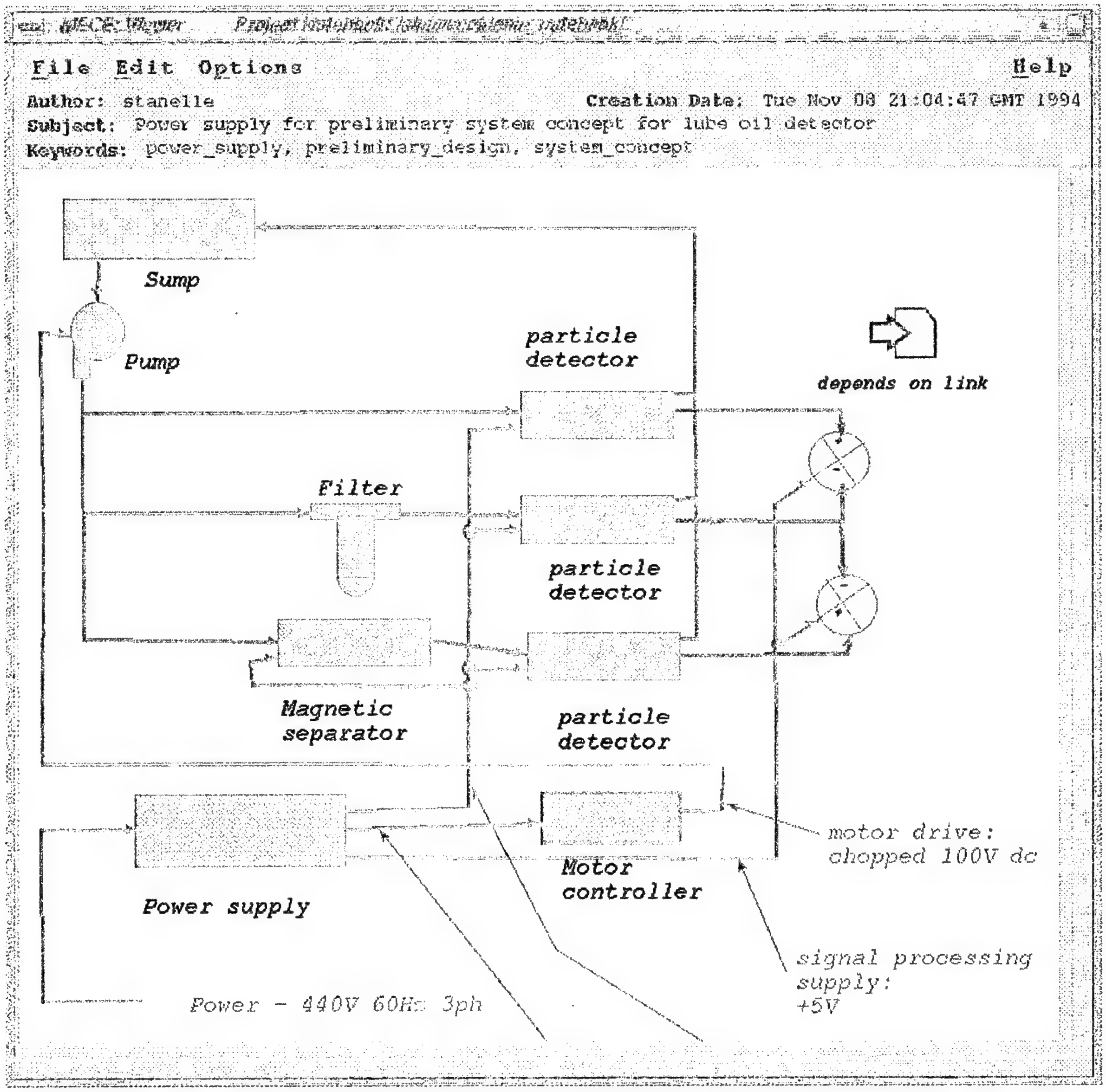


Figure 2.6. The MECE viewer displays information entries

MECE has a client/server architecture. The server currently runs on a UNIX platform and maintains the project notebooks. A client runs at each remote site, and user requests for access to notebook entries are handled by the client which communicates with the server. Communication is accomplished with the HyperText Transport Protocol (HTTP), the communication protocol used by the World Wide Web (WWW).

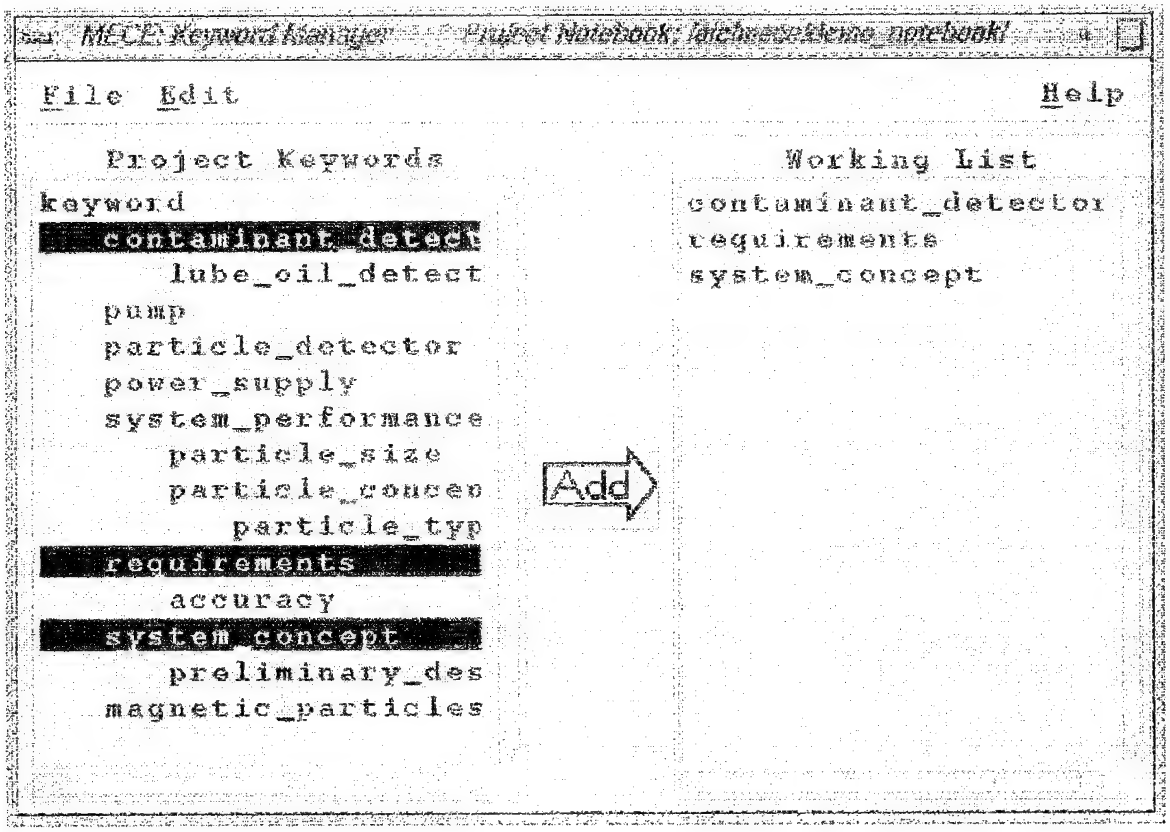


Figure 2.7. The MECE keyword manager maintains an official list of project keywords

The project notebook is currently implemented as a flat file structure. It consists of a set of MECE entry files, an index file containing header information for each entry in the notebook, several link index files, and a file containing the identification tag of the last entry published to the notebook. In addition, there are subdirectories for storing media files (images, audio, video) referenced by notebook entries. The search engine uses a query language defined for MECE to access entries in the project notebook. The flat file structure works well for small notebooks (less than a few hundred entries). A commercial database (e.g. ORACLE) would be preferred to support larger project notebooks. In anticipation of the future need to link to a commercial database, the search engine has been implemented in a modular fashion. Only a core set of methods that deal with constructing the actual query would need to be modified, substituting SQL (in the case of ORACLE) for the current query language.

Each entry in the notebook is represented and stored using an extended form of the HyperText Markup Language (HTML), the form used by the WWW. The MECE extension to HTML is referred to as ABSML (A Better Structured Markup Language). There are two reasons for using an extended form of HTML. First, MECE is required to be compatible with the WWW. Second, it was agreed that engineers are used to arranging sketches and text spatially in their paper notebooks. To maintain this capability in the electronic form of the project notebook, additional markups for HTML had to be defined that allowed for spatial arrangement of objects. For instance, HTML does not allow one to overlay text on top of an image, and ABSML does. In addition, HTML only

provides for text and image objects in a document, and ABSML provides for a much richer set of object types.

The MECE authoring tool is object-based. Each element that is created on the authoring canvas is implemented as an instance of a C++ class specifying the element type. For example, when the user records an audio clip an instance of the AUDIO class is instantiated. Each class type (SEGMENT, ELLIPSE, AUDIO, etc.) is a subclass of the meta class MECE_OBJECT, and in general, each type of object is referred to as a *mece object*. Each class type has a defined set of methods for manipulating objects of that class. Using the audio object as an example, each time the authoring canvas is redrawn, the draw method for audio objects is called. The objects contained in an entry are maintained as a list.

Mece objects can be copied and pasted between authoring tools or between an authoring tool and a viewer. Object identity is of course maintained. Several editing capabilities are also possible. These include move, group, and various transformations (reflection, scale, etc.).

From an implementation point of view, publication causes several actions to occur. First, the entry is assigned a unique identification tag, and the file in the notebook containing the id tag of the last published entry is updated. Second, the entry is indexed in the notebook based on the header information that was supplied by the author. This is done by appending the id tag and header information to the end of the index file maintained in the project notebook. Third, the link index files are updated to include link information for the new entry since, if the new entry contains hyper links to other published entries, this information needs to be available. Next, if the entry contains media objects (images, audio, video), the associated media files are copied from a temporary storage location to the appropriate subdirectories of the project notebook. Finally, the ABSML string representation for the mece objects contained in the entry is saved as a new entry file in the project notebook.

The main component of the navigator is the search engine. A search query string is constructed from the search parameters that are entered by the user. It is then passed to the search engine, which tests it against the header information for each entry in the notebook. In an early prototype of the navigator, the header information was maintained at the beginning of each entry file. This design proved to be quite inefficient for large notebooks since each file had to be opened and closed for each search The current implementation maintains the header information for all entries in a separate index file. Thus, no matter what the size of the notebook, only one file I/O is performed. This design change provided a substantial improvement in search time.

If the user specifies link information in the search query, the search engine will access the link index files that are maintained with the notebook. These files contain information describing forward and backward links between entries. There are four different types of links: "reference", "depends on", "see also", and "revision". The user performing a search has the option of specifying either all links or links of one of the above types, in either the forward or backward direction, or both. Thus, there are ten different link search files. Each corresponds to a type of link in either the forward or backward direction.

Both the keyword and personnel managers are implemented as special applications of the taxonomy manager. Both keywords and personnel records are implemented as special types of MECE entries in the project notebook. As a result, the

user may search for both keyword and personnel entries the same way in which the user searches for other entries in the notebook.

2.2 Proposed vs. Implemented Functionality

The functionality proposed in 1992 for the MECE system is shown in **Table 1**.

• The current MECE system represents a fully capable multimedia-based collaboration environment. It provides all the necessary data management tools (authoring, publication, & navigation) needed to allow development or importation and indexing of engineering notes, multimedia e-mail, and sketches; and the capture, editing, storage, indexing, transmission, and retrieval of snapshots. The authoring tool contains capabilities for easily marking up entries including intrinsic text and graphics primitives, audio and video clips, snapshots, and high-level annotations.

• Electronic mail is integrated with MIME to create multimedia messages using the Publish agent, to order to record the messages as permanent entries in the engineering notebook. E-mail is also used for automatic notification at notebook entry publication time. Recipients of MIME mail messages view the contents of the newly-published entry using a MECE viewer. This implies that, in the case of notification, one does not need to search the notebook for the new entry, since it is available as part of the mail message.

• While DIRCCE embodies several useful capabilities in support of configuration management, the system as a whole is not mature or complete enough for integration with MECE. For example, DIRCCE lacks a security mechanism for access control, and lacks other features we expect to find in a database with version control. As a result we are in the process of selecting an object database which has all the facilities that MECE requires.

• Voice and video annotation has been realized to different degrees. In addition to the graphical mark up capability described above, one can attach both voice and video annotations to MECE entries. Recording and including voice and video annotations in an entry results in the annotation being saved as an external media file within the project notebook. A standard, platform-independent format is used for storing audio annotations. Video annotations are stored in an MPEG format.

• The goal of real-time collaboration has been realized in the form of the audio/video/notebook (AVN) conferencing component of MECE. Using this

Proposed Capability	Implemented Capability
Collaboration with text and graphics	First MECE prototype
Integrate e-mail messages with MIME	E-mail, Automatic notification
Storage and retrieval with DIRCCE	Integrate commercial database
Enhance MECE to support voice annotation	Second MECE prototype
Enhance MECE to support digital animation	Commercial software
Enhance MECE to support video annotation	Third MECE prototype
Enhance MECE to support video conference	Real-time collaboration, shared authoring
Workspace integrated applications	Hot links
Circular buffer	Commercial software
Digital Animation	Commercial software

Table 2.1. Proposed and implemented MECE system functionality

conferencing capability, a team of geographically-distributed engineers can jointly share an authoring tool, annotate an entry, and publish the entry to the project notebook while in audio/video conference.

• To meet the goal of integrating external applications, the hot link capability was developed. This capability allows one to create an active link to an external application and to associate a data set with the application. Users may then include snapshots of the results that were generated by an external tool (e.g. CAD) in their MECE entries. A snapshot is static in that it can not be manipulated in any way, but the hot link capability allows the user to include a link to a dynamic model consisting of the tool and the corresponding data set.

• The circular buffer concept, which allows one to save part of a conferencing session and to play it back later as part of a MECE entry, is not currently supported. To realize it implies synchronization of audio, video, and mouse movements. Based on an estimate of the amount of time required to implement a circular buffer in MECE, it was decided to integrate software from
a commercial vendor. This will take place in the near future.

• Digital animation allows one to create a movie and to include it as an object in a MECE entry. Silicon Graphics, Inc.(SGI) already has adequate commercial software bundled with their operating system. As more commercial animation software is developed for other UNIX platforms, the goal is to integrate that software with MECE rather than reinvent animation software of our own.

2.3 DIRCCE

The Distributed Revision Control for Concurrent Engineering (DIRCCE) system was developed at the Lockheed Artificial Intelligence Center in response to the need for a shared database under configuration management control. DIRCCE has a client/server architecture, where the server maintains the consistency of the shared database using UNIX's Revision Control System (RCS). A client runs at each remote site, and user requests for access to data are handled by the client which communicates with the server.

To decrease access times, a local cache of relevant database objects is maintained at each remote site. DIRCCE provides for the automatic update of the partial copies of the database at each remote location. The first access to a design object at a remote site causes that object to be retrieved from the central database. Subsequent accesses to that object by designers at the same site are purely local. If the object is modified at another site, the remote site receives automatic electronic notification, and the object in the local cache is updated. The system also provides other services, such as versioning, generally associated with a configuration management system.

2.3.1 DIRCCE Applicability to MECE

The MECE system design called for rapid access to and sharing of not only informal MECE entries but also actual design data for the purpose of viewing and modification. Additionally, since the MECE system is to be deployed within projects whose team members will often be geographically distributed, access to data is needed over a wide area network (i.e. Internet). Furthermore, design history information and a notification mechanism for actual changes to design data, similar to the project history and notification schemes for informal MECE entries, were also required. Currently DIRCCE meets these needs very well for small design teams working on UNIX platforms. On the other hand, it does not seem that DIRCCE will scale very well, and

full PC and Macintosh versions of DIRCCE are not available. As a result, while the capabilities DIRCCE provides are both needed and useful, it seems clear that a commercial database might better meet our needs.

2.4 Design Rationale

The MECE project notebook is to serve as a shared, collective memory throughout a project's lifetime. At any one moment in time, the notebook contains all the design rationale and results of informal, daily collaborations that have occurred on the project to date. To enable this continuity a conscious decision was made early in the design of the MECE system that the project notebook should remain monotonic. That is, users are always able to create new entries in the notebook, but once an entry is made it cannot be deleted. Allowing entries to be deleted from the project notebook would destroy its integrity. The chain of rationale leading up to later decisions would be broken. and these decisions would become inconsistent with their origins.

For the same reasons, republishing of existing notebook entries is discouraged. Exceptions include correcting spelling errors, modifying entry keywords, and correcting other entry header information that would affect indexing and retrieval. Although there currently are no limitations on what information can be modified during republishing, it is expected that engineers will use this feature with discretion.

A key objective of the system architecture was to maintain an open environment, and his was met by the development of an object based architecture.

MECE is an informal tool used for recording all forms of collaboration, including daily discussions, rough sketches, and detailed designs. On the other hand, MECE is not intended to be another desktop publishing tool. Rather our goal was to promote design function by providing an environment that enables quick and easy generation and/or assembly, mark up, and exchange of multimedia information. MECE is similar to electronic mail in that it is an informal collaboration tool that can be used easily, often, and freely.

2.5 Some Results of DICE 5

This has been an unusual R&D contract in that the product, in this case MECE, was put into practical (and critical) applications before the contract was completed. It was used to support the early design phase of a major concurrent engineering contract in Simulation Based Design, and to support aircraft design at the Lockheed Skunk Works. Its versatility was demonstrated when it was equipped with templates to support Quality Function Deployment.

References to MECE applications to SBD are found throughout this report. An overview of the Skunk Works applications follows, copied from an informal note:

<u>JAST</u>

"We created a notebook based on a trade study performed on a component designed for the JAST program. The particular component, an inlet duct, was designed to demonstrate Affordable Aircraft Acquisition (AAA) methodologies. Included in the notebook are the aircraft surfaces, the finite element analysis (FEA) results of two different configurations, the actual designed part, and photos of the completed part as well as the testing procedure."

JASSM

"Another notebook was begun for the JASSM program. In it we included hand drawn sketches and diagrams as the first steps in documenting the design. This proved to be quite popular when presented to the designers. They seem to appreciate the ability to employ a pen and paper as their primary notebook tool."

This note included a list of proposed enhancements to MECE, to support the particular needs of the Skunk Works users, and endorsed the overall system.

An interesting application to Quality Function Deployment (QFD) was realized by adding templates to a MECE notebook. This was useful in itself, but also established that MECE could support complex applications within its straightforward environment. The QFD writeup follows:

"MECE provides a facility for performing QFD across a networked system. Traditionally, team collocation was vital for meaningful results from QFD. With the MECE QFD facility, anyone with access to a networked workstation may participate fully in the QFD process. The QFD facility provides participants with a rich set of tools for expressing opinions, for providing ratings, and for responding to dialogue at each stage of the QFD process. The idea is to retain as much as possible of the discussion that occurs in the traditional QFD process while also allowing the process to be both documented and performed asynchronously.

To implement the QFD capability, the MECE template mechanism was extended to provide additional templates for each of the rounds of QFD. For example, the template for establishing importance rankings contains space for rating each customer attribute along with commentary as to why a particular rating was given. Because of the multimedia capability of MECE, this commentary can include not only text but also graphical annotation and hyperlinks to relevant MECE documents such as market research and drawings from past projects.

The MECE QFD facility distills individual submissions into a group summary. The group summary includes average ratings and a compilation of commentary. Individual team members can review the group summary and can then change their original ratings and opinions, if desired, based on information from the group summary. Immediate discussion via electronic mail, phone or personal conversation can of course be used throughout the process. A final group summary is created, and the process moves on to the next round. At the end of the QFD process, a graphical House of Quality is generated."

It is planned to use this template facility to add a Requirements Manager and a Workflow Manager to the MECE environment. (See Chapter 4).

2.6 Lessons Learned

There are two major design phases for any complex interactive system. The first is based on perceived requirements, and the second on those requirements that only become clear when the system is in practical use. If the initial design is inadequate the prototype may be rejected before it can provide a framework for its evolution, or be very difficult to change to meet the second set of requirements. MECE passed these tests with flying colors, in that the prototype was readily accepted for practical use and the system has proved easy to evolve.

Over the course of MECE testing and evaluation, several interesting lessons have been learned. First, as expected we encountered cultural barriers which have hindered the deployment and acceptance of MECE. This problem had been anticipated as detailed in Chapter 3.

Several problems came about as a result of conflicts between the MECE philosophy and expectations which users had as a result of experience with desktop publishing and word processing environments. Since the authoring (or more properly entry assembly) tool has strong parallels with desktop publishing, it was more subject to this sort of problem than, say, the navigator where there are fewer pre-defined expectations. Unfortunately, these expectations often appeared in conflict with features which achieved some benefit. This lead to the unappealing choice of arrogantly ignoring the users' feedback (we the designers know best) or giving up some desirable feature because the users' suggestions did not consider the full extent of the tradeoffs involved. A standard example of such tradeoffs are verbose commands which are prized by new users and despised by the same users after they have become familiar with the system.

Some examples of problems of this sort include:

1) Limited number of fonts and type faces
2) Lack of features for controlling format of printed output
3) Lack of features for making entries of "vu-graph" quality and appearance
4) Lack of features for controlling format of text
5) Use of "click to initiate / click to terminate" gestures for drawing, as contrasted to "click and drag" gestures.

We had anticipated a number of cultural problems associated with the "permanent" aspect of the notebook, causing people to delay publishing ideas until they were "ready", and a natural desire to be able to retract or modify things which were ultimately proven to be errors. However, we continue to believe, using the Skunk Works as an example, that these attitudes do not really have a place in effective team environment and must therefore be changed by management by providing a good cultural framework, and should not be catered to by MECE.

However, another problem arose when people wanted to create entries using desktop publishing tools and embedding them in MECE as snapshots or hot-links. Initially we thought that our authoring environment lacked expressive power in some vital way, but examination indicated that in all cases the intellectual content of the entry could have been expressed as quickly (often more quickly) in MECE. The problem was that the result was not as visually appealing ("looked sloppy") because in the interest of providing an easy to use interface, MECE failed to provide a number of features for creating symmetrical objects, snapping lines to objects or grids, and performing other operations which make graphic images attractive. In many cases the features in fact did exist in MECE, but the user was not familiar with them, so the "deficiency" was in user training rather than the system itself.

3.0 TEST AND EVALUATION

MECE has been in test and evaluation (and refinement) since the first prototype became usable. It was and is a moving target, in that shortcomings discovered in use have been and will continue to be promptly overcome, and opportunities for improvement were and will be exploited. The term "evaluation" must refer to MECE at a certain date, in this case 1 March 1996. The current list of enhancements is long enough to drive development for a year, and there is no reason to believe that the evolution of the MECE environment will stop.

The bottom-line evaluation of an interactive system has two aspects: whether it improves productivity in the sense of quality and quantity of work, and whether it is accepted by the user community. If it is felt to be inadequate, or too inconvenient, or if its learning curve is too steep, it will not be adopted whatever its intrinsic merits. The first task is to gain acceptance so that the system can evolve as guided by the user community. MECE has passed this T&E threshold.

3.1 Overall Approach To Test And Evaluation

The perennial problem in deriving an objective measurement of increased productivity is to find an appropriate documented test case to serve as a baseline. This implies 1) that the new system is introduced into an environment where either a comparable system exists and has been evaluated, or that productivity measures have been derived from observing the target user population at work, and 2) that the work products and processes are comparable, that is that the new system does what the old one does, but presumably better. Neither of these assumptions is valid in the environments where we have tested MECE. It does not supplant a similar "system" nor can it be compared with a documented test case, partly because the kind and degree of collaboration that MECE enables was not previously available. These were limited to normal meetings, e-mail, regular mail, and telephone calls.

Measuring acceptability is easier than comparing performance, granting that it is possible to install the system in the kind of environments it is planned to serve. This has been accomplished.

As a result, and as is usual in the evaluation of a system which breaks new ground, we use case studies and qualitative metrics. Seven metrics were cited in Lockheed's DICE 5 proposal. The hidden assumptions were that 1) there would be a substantial number of working prototype systems at the time the evaluation took place, and 2) that there would be a working system with which MECE could be compared. In practice, neither situation occurred.

It <u>was</u> possible to observe the amount of use of MECE in several test environments, and assert that it supported design and design documentation in all three, but there is no baseline to substantiate a claim that the design or documentation is 50% better, or 35% more complete, or six times easier to capture and record - better than what?

On the other hand, if MECE had not been available to SBD it is probable that little organized design documentation would have been circulated or captured, with the corresponding lack of context for intelligent design, so the use of MECE was extremely valuable.

3.2 Pilot Site Application: Two Case Studies

3.2.1 Simulation-based Design (SBD)

MECE was used in the conceptual and ongoing preliminary design stages of SBD, and will be used in the detail design. This case did not test MECE distributed collaboration because the design team was small and collocated, but did provide a convincing demonstration of the power of MECE for information sharing in an evolving project notebook. The notebook structure of filterable nodes and links, provided a powerful and flexible way to develop, document, relate, and focus on design issues and decisions. The SBD design notebook is large but manageable, and constitutes an invaluable "living" reference for project development.

The MECE Authoring (authoring, importing, capture, indexing, and "publishing") and navigation (filters, keywords, spatial linking and display) tools were tested and refined over a period of months in this application.

Conclusions: this experience provided a new and far better working environment for collaborative design than any that the team members had experienced before. It is not easy to measure the increase in design productivity (in particular in design quality) due to increased shared understanding of the design issues common to several disciplines, but this may have been the single greatest advantage of the shared notebook approach.

3.2.2 Lockheed Advanced Development Company (Skunk Works)

MECE was deployed at the Skunk Works as a concurrent engineering tool for the design of the Lockheed version of the JAST VSTOL fighter-bomber. (See Section 2.5)

3.3 Metrics Used

The Lockheed DICE 5 proposal cited six critical performance parameters to be evaluated using seven metrics. The performance parameters and a capsule of the user consensus as to their realization in the three test user environments are:

• *Ease of Collaboration.*

-The SBD experience established that it was easy and natural to use the shared notebook.

• *Ease of Storage and Retrieval*

- This was established by experience in all three cases

• *Timeliness of System Response*

- Timeliness is a highly subjective issue. Some elements of system response seem laggard, in particular the act of "publishing", until it is realized that a complex process is hidden behind an apparently simple request. This will only be overcome by more powerful hardware, and in the meantime users will be educated to expect it. Most responses seem quite acceptable, and normal in a client-server system

• *Ability to Handle All Types of Information*

- This refers primarily to the variety of media MECE is designed to use. The SBD application, so far, has used text, graphics, and captured (bit mapped) information.

• *User and System Errors*

- No statistics have been collected. The system users have accepted the user and system error rate as normal for a prototype, and continue to use it while detected errors are corrected. Overall the error rate has decreased, despite the fact that new facilities are being added and old ones improved constantly.

• *User Acceptance*

- This has proved to be high everywhere. Every user group has suggestions for improvements and new features, but no plans to modify the basic system.

3.3.1 The metrics, and the results of their application, are described in more detail:

• Direct use measurements.

These were to evaluate timeliness of system response and user and system errors. These measurements were not made objectively because the system was in a constant state of evolution, and, for example, many system errors were not repeatable because they were promptly corrected.

- The subjective results indicated that there is one annoying delay, in "publishing", which is due to the large amount of processing this task demands, and is probably inevitable; and that user and system error rates were both high but rapidly decreasing, as to be expected in a new and developing system. As usual, the introduction of new features increases both error rates until the features become familiar and mature.

• Categorizing and counting entries.

This metric was intended to evaluate ease of collaboration by noting the rate of growth of the number of entries, and the amount of discussion relating to specific topics; ease of storage and retrieval; ability to handle all types of information, by examining entry types; and user acceptance, by noting how many people use the system on a regular basis:

- Since all entries are time and date stamped it is easy to plot the growth of the 240+ entries in the SBD design notebook, but the rate of growth has been affected by external situations (such as the temporary allocation of personnel to more "urgent" matters) to such an extent that the apparent rate is not characteristic of a normal project development. The result is that one contributor, the chief designer, has a fairly steady input to the system, and most of the others make heavy but sporadic use of MECE. An important advantage of MECE in such an environment is that personnel returning to an interrupted task can easily catch up with their previous status and intervening developments, and quickly resume productive work.

- It was noted that, as expected, clusters of entries appeared around difficult topics of general interest, and that MECE served as an excellent forum for the discussion of these problems and evolution of solutions. Some other topics were developed in a linear way by a single author and with no dissenting voices. These were characterized as either "settled" by the author, or of more specialized interest.

- almost all the entries were "pure" text, and the rest were a combination of text and graphics. Since the SBD design team was collocated there was little incentive to use communications tools, except for the automatic e-mail notification of new entries. No one felt that it was worth the effort to animate their entries, and this was probably because they represent the conceptual and preliminary design of a computer system, which is not a good subject for animation.

- with one exception, all those members of the SBD design team who had access to an appropriate UNIX system used MECE whenever they had a contribution to make to the design process. The exception was a design team in constant touch with a large number of remote subcontractors to all of whom it was impractical to provide (and support) MECE installations.

• Fraction of design team members who use the system for a given purpose.

- In the case of the SBD design team the fraction was about 80%, and the other 20% did not only because it was not practical to set up and maintain a large number of remote sites. (See previous paragraph). This indicates that it was the preferred mode of collaboration for all design team members, which testifies to it ease and user acceptance.

- This metric is also intended to evaluate the ability to use all types of information, but in the case of SBD early design the users did not show interest in any but text and simple graphics.

• Completeness of start to finish problem solving applications.

This metric is intended to measure ease of collaboration by determining whether users tend to abandon MECE before design decisions are finally made, and the answer is clearly negative. There are pending design issues in the SBD design notebook, but they have not been abandoned, and will be addressed when other design team members have the opportunity to contribute to their resolution. On the other hand, almost all the issues of immediate interest have been resolved, that is, have arrived at a generally accepted solution. This fact indicates ease of collaboration, ease of storage and retrieval, use of all relevant types of information, and above all user acceptance.

• Decreased use of primitive alternatives.

This metric was to evaluate ease of collaboration and the use of all types of information, to show the extent that MECE subsumes conventional techniques for collaboration.

- Since there was no baseline measure of the use of alternative means of collaboration in similar tasks no comparisons were possible.

• User survey.

This "metric" provides a subjective evaluation of all the performance characteristics.

No formal survey was made, unless the report from the Skunk Works could be considered to be a survey, but the MECE development team was in constant communication with the users and aware of their reactions. When the fragility of any prototype is taken into account, and most users understood this problem, the system was both acceptable and used. There is a host of small issues to be resolved and local improvements to be made, but this is a normal and healthy situation: the user community wants MECE, and they want it to be more perfect; and the developers have the same goals.

• Number of technical disciplines using the system.

The point of this metric is to show that MECE was used successfully as a concurrent engineering tool.

- Since the SBD early design task represents a very limited slice of the product (SBD) lifecycle, and SBD is a software system, it cannot be considered to be a typical

(hardware) product development effort. In any case, several different engineering and computer science specialties were represented, and MECE certainly promoted their interaction. All the desired types of information were available and used, and the designers found the notebook to be a highly satisfactory forum for different viewpoints.

3.4 Cultural Findings And Premises

An CSCW environment like MECE tends to cause cultural change because it changes the relationships between the players in the game, and provides tools and a shared context for group work. This meets a certain automatic resistance, if only from inertia, whatever its appeal or value. Much of this resistance is due to faulty perception of what MECE has to offer, most of the rest is due to its unfamiliarity, and both can be overcome by the right kind of exposure.

• Perception - <u>MECE would not offer anything to an environment that does not require collaboration</u>.

Part of the current and ongoing change in industrial practice is in the importance given to collaboration. Practitioners of CE give it extreme importance, but organizations which feel they are self sufficient and/or are preoccupied with a single phase in the product lifecycle may disagree. Also, distributed collaboration does not seem important to a small facility where the users are collocated, or where responsibilities can be allocated to make engineers almost independent of each other

.- <u>Actuality</u> - Even assuming that the above objections have merit is some cases, MECE is a superb working and documentation tool for individuals, as well as groups.

• Perception - <u>Collaboration may seem desirable, but it is a small part of the current pattern of work and is not cost effective to implement</u>

Collaboration can be a difficult, time-consuming, and unsatisfactory process. Lack of a shared context for discussion, endless meetings with too many and/or the wrong specialists, telephone calls documented with scribbled notes, and vanilla e-mail exchanges can provide too little relevant, usable information to be worth much effort. The result is the effort is not made.

Lesson learned - The first step is to make collaboration meaningful, and the next to make it easy.

- <u>Actuality</u> - MECE provides for 'tailored' collaboration and exchange of large 'chunks' of information, based on the mutually accessible context of a project notebook. Consultation via conference or multimedia e-mail is focused on the user's task at hand. The information exchange can include voice, text, diagrams, sketches, even animations so the content can be arbitrarily large and complex, and it takes place in an environment where all the current supporting information is a few hypertext jumps away.

MECE collaboration is useful, and it is also easy. If the circumstances are right, video/voice teleconferences can be arranged in a moment, otherwise they have to be scheduled in the sense that telephone call must be scheduled. The effort in multimedia e-mail is comparable to e-mail with text enclosures - the extra work is in producing or locating the enclosures, but the effect can be much more informative

• Perception - <u>interactive computer systems are expensive, hard to learn, full of bugs, and hard to maintain.</u>

New software systems invariably suffer from some degree of these problems. Systems like MECE, if developed to the shrink-wrapped stage by a commercial vendor, have to be relatively expensive to recapture their costs.

Until an interactive system has been in use by a diverse group, and for some time, its interface will be less than ideal. Its functions might be transparent to its developers, but only a small percentage of the potential users may be familiar with the developer's environment, so the interface must be changed to be acceptable to a wider range of users. In the meantime good initial design allows the system to survive until it can be perfected.

All new and evolving systems have errors, and the best that can be done is to assure that they are minimized, do not have dire consequences and that they are quickly corrected.

It is generally too difficult, and undesirable, to perform in-house maintenance of a system which is to be kept compatible with those of other users. On the other hand, problems arise when the vendor is slow or reluctant to respond to perceived and real errors.

- Actuality - MECE is not free of these problems, but they have reached a manageable level.

MECE, as it stands is very inexpensive to acquire. It is free, although with no guarantees and no free maintenance. Assuming MECE becomes a commercial product, it will still be relatively inexpensive because the difficult period of prototype development has been financed by ARPA DICE.

The MECE interface is relatively easy to operate, and the MECE agents work in a comprehensible way that makes their mastery easy. There is work to be done to make some of the operations easier, and it is under way.

There are, and will be 'bugs' in MECE, and they will diminish in number and effect as the system matures. The important criterion is their effect on MECE system usability, which can be defined as loss of data (not a problem, except for transient data), inaccessibility of an agent (very rare), system faults (mostly due to the operating system and communications), and apparently random faults which defy easy classification (rare).

MECE has proved to be comparatively easy to maintain because it has a component architecture rather than a monolithic structure.

4.0 Recommendations for Future Work

Figure 1 is a high-level object model of the MECE distributed collaboration environment. It will be used to illustrate what has been done to realize this capability, and what is left to be done in the future.

Our initial approach was concerned with the basic issues of sharing information and activities, which are arguably the core of collaboration. Lockheed did not ignore the important aspect of shared direction but did not give it the same priority.

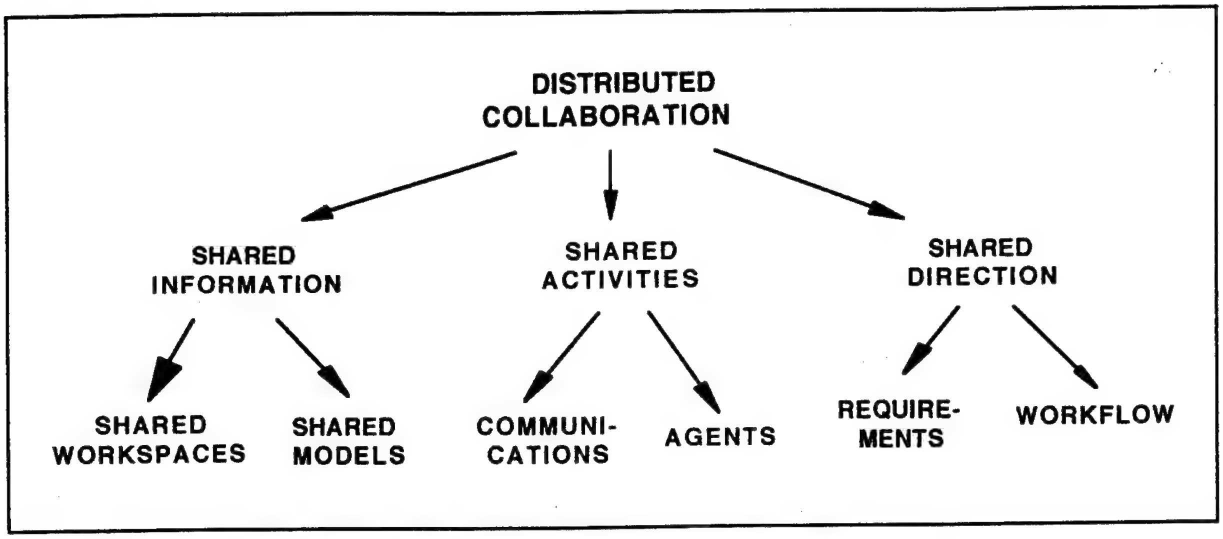


Figure 1 - A top-level requirements model for MECE

Some of the tasks to be undertaken are motivated by the desire to complete the system as outlined in Figure 1, and others (in particular improvements on prototype implementations) are motivated by user experience and evaluations.

4.1 Shared Information

Shared workspaces are represented by MECE group and project notebooks, and they are already in use. There will be refinements in the way notebooks are accessed and maintained, but only after more experience has been accumulated and documented.

Large scale and detailed shared models are created and maintained as companion activities to MECE, which is engaged in motivating and guiding their development and annotating and recording them, their relationships, and their lifecycle. In this regard the current MECE capture, annotation, storage, and retrieval techniques are working well. There has been some interest in enhancing the text and graphics tools inherent in MECE to make it a more flexible environment for conceptual modeling and design, but this may be postponed until the possibilities of OpenDoc (which would permit the use of any generally accepted tool) have been evaluated.

4.2 Shared Activities

Augmented communications capabilities will be purchased and integrated as their technologies evolve and they become more affordable. Video conferencing is becoming more useful as common carrier data rates increase and lossy compression techniques mature, and the small and jerky screen images are replaced by larger ones with more realistic motion. Better platforms to develop multimedia mail are becoming available. An

example is Adobe Acrobat, which provides a mature and friendly environment for creating/reading hypertext multimedia documents. Computer voice communications are becoming more mature, as there are a growing number of applications supporting "telephone by internet".

The major agents, Authoring (generation, capture, annotation, conferencing, e-mail, indexing, publishing/storing, and notification) and Navigation (multi-mode retrieval, display) have proven to be very effective. As mentioned above the choice of text and graphics toolset could be opened by the use of OpenDoc, but there doesn't seem to be any reason to change the publication procedure - except to find ways to make it faster. As experience accumulates these agents will naturally evolve into more polished, easier to use versions but their generic functionality seems to be established.

4.3 Shared Direction

Two MECE agents have been designed to support shared direction. The Requirements Manager deals with application and development of product requirements, to help engineers do the necessary work on the right topic. The Workflow Manager helps to coordinate their efforts. It is planned to implement both as templates, to provide a semi-formal framework for their users.

These agents were designed to support other projects (Simulation Based Design (ARPA) , and Simulation Assessment Validation Environment (JAST)), with the intent to integrate them with MECE. Their general nature makes them both project independent and suitable to round out the capabilities of MECE as a total collaboration environment.

4.4 Infrastructure

There are several emerging opportunities to make MECE more effective and even easier to use. OpenDoc promises to provide a platform independent document-centered system which might be an ideal host for MECE. When the UNIX version of OpenDoc becomes available this possibility will be considered in depth. In the meantime Acrobat might be considered as an authoring tool, but only if it were available on a wide variety of UNIX platforms. It is already in use on the Macintosh and Windows systems.

MECE currently supports WWW pages as notebook entries, and the possibility of using JAVA to make them interactive would enhance their value. Another potential application of JAVA would be to launch external tools, which could provide their initiation scripts to the MECE notebook user or application scripts. This could make the Workflow Manager a true process controller.

APPENDIX A - MECE REQUIREMENTS

1.O Introduction

MECE is a CSCW system intended to support concurrent, hence collaborative engineering. As such it shares the generic requirements model shown in Figure 1, where all the arrows are read "depends on".

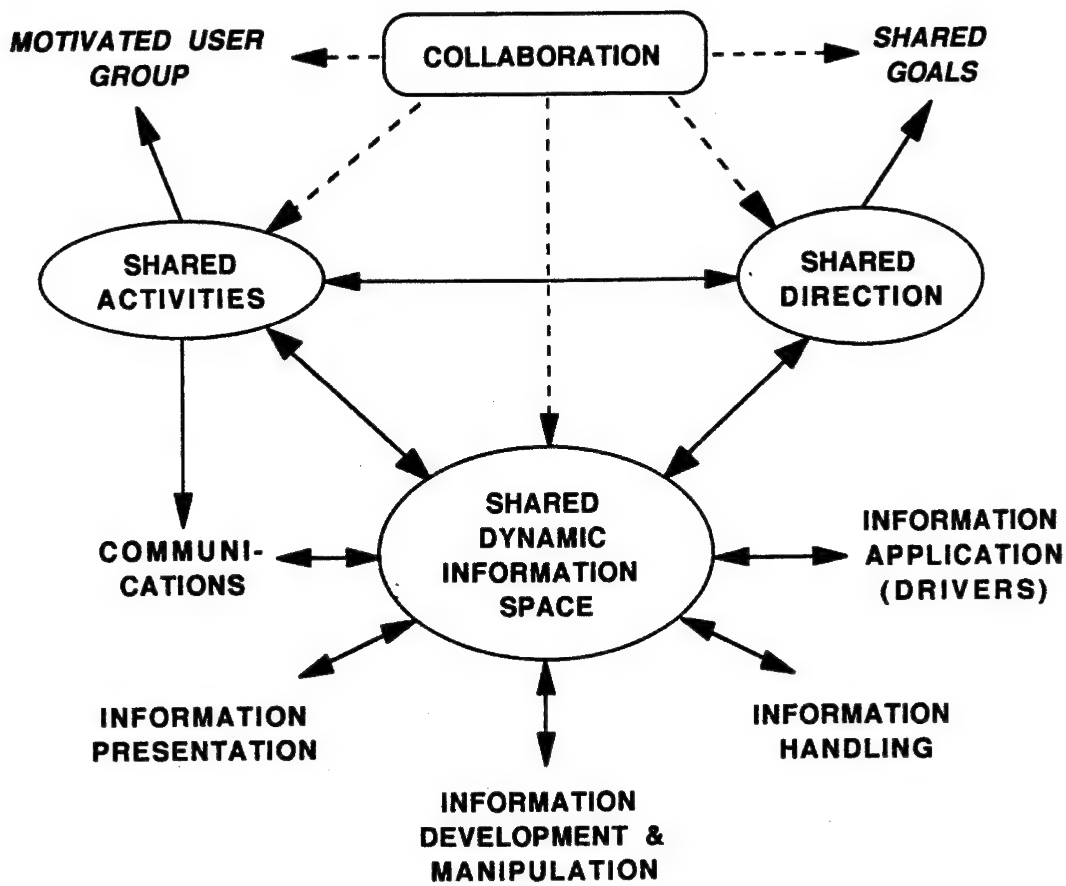


Figure 1 - A Generic Engineering Collaboration Requirements Model

Figure 1 emphasizes that there are two elements of collaboration which are independent of any system built to support it: if the intended collaborators are not interested in working together, or if there is no agreement as to the goals of the effort, attempts to collaborate will fail.

The three general requirements below characterize any collaborative system:

1) Shared Activities include the various phases of design, the development of an advertising campaign, specification of manufacturing and maintenance procedures, etc.,. etc. Their cooperative execution depends on a shared direction for the work and the shared information arena where development takes place. Informal communications take place in parallel with as well as in support of the "public" work

2) Shared Direction provides a common understanding of the nature of the task and individual responsibilities in its performance. It is documented by requirements derived from the goals of the task in hand and decisions which realize these goals, and which generate new requirements.

3) the Shared Dynamic Information Space is the arena where the evolving product is generated (or imported), displayed, altered, annotated, stored, and retrieved. It may be public, restricted to the current group, restricted to a team within the group, or currently to an individual, but its intent is to provide a living (and active) document to support collaborative activities by providing access to the whole situation (past and present) and providing tools for its further development.

The five requirements below are derived from the first three, and they include the needs of a distributed concurrent engineering system. They could be relaxed for less demanding applications.

- Communications are inseparable from concurrent engineering (and other) collaboration since we must assume collaborators may be dispersed. In our design the conventional forms of communication are augmented by "whatever works", including the WWW.

- Information Presentation is a general requirement, which is realized for MECE by a wide range of familiar forms such as text, graphics, 3-D models, audio, and video clips, and new and more powerful forms, up to and including animated virtual prototypes and immersed virtual reality.

- Information Development and Manipulation refers to the information tasks and their tools. Our approach to the requirement includes tools for writing, association by hyper-links, drawing in two or three dimensions, modeling in three or four (including time), attaching audio and video annotations, etc.

- Information Handling is the process of information storage and retrieval, to which MECE adds notification of change, versioning, etc.

- Information Application is not always required for collaboration. In an engineering system it is the capability of the overall system to interact with the external world. These interactions may be in support of the local shared development activities or may govern external processes.

The requirements model will be used to organize the discussion of MECE-specific requirements and their realizations. The centerpiece of the model is the Shared Dynamic Information Space and MECE implements this concept and requirement.

2.0 MECE Design Requirements

System Goal Requirements

- The capability goal of MECE is to help dispersed engineers create, develop, associate, store, exchange, and view design related MM information.

- The system can be distributed over space

- information shall be permanently stored and easily available

- Ease of use is essential

- MECE is an evolutionary system, and must be capable of improvement, growth, and change for the foreseeable future

- MECE is to be usable on multiple platforms.

2.1 Shared Activities

Shared activities are supported by communications and SDIS capabilities.

- The system communications facilities shall support user team interaction with the SDIS and its tools.

- Some communications, particularly those with colleagues without MECE facilities, will bypass the information space. Their content shall be captured and included. For example, E-mail shall be captured and treated as SDIS entries.

2.2 Shared Directions

Shared directions shall be supported by public requirements statements and agreements as to task responsibilities

- Systems and component requirements shall be part of the design documentation

- Design decisions and the resulting requirements shall be accessible to all team members

- Interested/concerned team members shall be notified of new or changed decisions and/or requirements

- MECE shall have a facility for the assignment of responsibilities to team members

2.3 Shared Dynamic Information Space (SDIS)

The information space shall have the characteristics of a shared notebook, or "blackboard", on which entries are posted to be seen, commented on, and used by the working team.

It will be desirable to have several levels of sharing; from private, to team, to group, to official (public) workspaces between which information is promoted as it matures.

2.1.1 Information Presentation

Capabilities:

- to display and/or activate the shared entries
- associative browsing

Entries in the shared space shall be:

- Accessible: for examination, copy, annotation, further associations,
- Permanent: to maintain a history of discarded as well as accepted approaches
- Expressive: to provide appropriate representations for the task at hand. This implies multimedia, including "virtual" representations
- Associated: linked to other entries using a variety of relationships
- And may be "live": to activate representations, messages to colleagues, links to other entries and to external resources,

2.1.2 Information Development and Manipulation (IDM)

Capabilities for IDM include

- Generation and alteration of text, graphics, and multimedia
- Communications to consult colleagues, and
- Capture of resulting information exchanges
- Annotation and linking of entries
- On-line help to describe the use of the IDM resources
- Templates to facilitate the generation and maintenance of specialized information

There shall be no limitations to the number of MECE tools that can be open.

2.1.3 Information Handling

Information handling shall include capabilities for information dissemination, indexing, versioning, storage, associative browsing and retrieval:

- Notification of interested parties on new relevant entries, including annotations on previous ones
- Means for associating indices (such as keywords) with entries to facilitate their identification, categorization, and retrieval

- Retrieval of individual entries and sets of entries by combinations of indices

- Following associative links between the entries in a workspace to find entries of special interest (see Presentation)

- Obtaining agreement to promote entries to "versions" of authorized documents, labeling and controlling versions

- Placing and maintaining collections of entries (e, g, "notebooks") in accessible storage, and retrieving them on demand

2.1.4 Information Application

This includes capabilities to invoke and interact with external communications, software, and hardware systems.

APPENDIX B - MECE DESIGN NOTES

1.0 Introduction

This document describes the design of the capabilities needed for the MECE system to perform as required. First it descrbes the operation of a global clipboard-like exchange mechanism selected to be the center of the MECE system design. Next it describes each of the MECE activities accessible from the exchange mechanism. These activities include authoring and navigating multimedia (MM) entries in a project notebook, retrieving MM mail, obtaining help on a MECE function, accessing external applications, and initiating a virtual conference and possibly using a shared application.

2.0 Nucleus

The nucleus of the MECE system implements the Shared Dynamic Information Space described in Appendix A. It contains a global clipboard-like exchange mechanism that allows MECE activities to communicate. All the normal editing functions are present and associated with the exchange mechanism, which will always be active. For example, MM elements may be selected from one MECE activity and pasted into another MECE activity via the exchange mechanism.

2.1 Clipboard-Style Interactions

The MECE exchange mechanism supports the transfer of MECE MM elements between MECE activities. Selections may be made by displaying the exchange clipboard and selecting items, and currently-selected items are the subject of cut-and-paste operations. Selection is achieved by marking individual elements, grouping elements, chained selection, or selecting "all elements".

To control storage requirements, the user may define the length of the history to be saved (i.e. the number of elements to be retained by the exchange mechanism). The user may also set other properties at the system level, which displays the exchange clipboard.

2.1.1 Edit modes and gestures

Drag/Drop Metaphor: The drag and drop metaphor is achieved by mouse clicks. The first click selects a MM and selecting "copy" moves the element to the exchange clipboard. When "paste" is selected he second click pastes the most recent entry on the clipboard to its new location.

Cut/Paste Metaphor: Multiple cut and paste operations may be accomplished by using multiple mouse clicks to select multiple MM elements. Choosing "copy" will copy the selected items to the exchange mechanism where they will remain selected. "Paste" will copy the current selections retained by the exchange mechanism to the target. In addition, clicking them in the exchange clipboard will open the current selections to be altered.

2.2 Identifying MECE Activities

To distinguish MECE activity windows from other windows in the windowing environment, a special border will be used for MECE windows. It has been suggested that MECE windows be bordered with a sequence of 'MECE' text icons.

3.0 Shared Activities

3.1 Authoring

3.1.1 Creating atomic elements

MECE will support the creation of the following atomic elements:

- Sketching including text, line, ellipse, arrow, and freehand(mostly for annotation)
- Snapshot
- Import bit map
- Record/Import animation
- Record/Import audio sound bite
- Record video(external camera)
- Record "simulated video" of screen(screen grabber)
- DIRCCE hot links to drawings, data files, etc.

3.1.2 Arranging atomic elements

The MECE authoring tool will support the arrangement of atomic MM elements in three ways. First, an entry may be arranged by laying out elements spatially using select/move and cut/paste actions. Second, atomic elements may be overlaid on other elements in the entry. Finally, atomic elements may be arranged temporally by cutting and splicing animation sequences and sound bites.

3.1.3 Templates

The authoring tool will contain special-purpose templates. These templates will provide a preformatted entry and hints for the proper content of an entry of the type chosen. MECE will also provide a default set of templates which may be edited/augmented according to the desires of the project. Examples include templates for decisions, requirements, issues, ideas, design definitions, meeting announcements, and job assignments.

3.1.4 Moving around in an entry

Authored entries that are larger than the drawing canvas will be scrollable in both the horizontal and vertical directions. In addition, a text search capability will be offered initially with possible future expansion to speech and geometrical pattern recognition. The idea of providing bookmarks has also been suggested.

3.1.5 Publication of authored entries

Publishing a MM entry is accomplished in three steps. First, destinations for the published entry are specified by the user by choosing multiple destinations from a list of candidates, which may include the project notebook. Then header information is associated with the entry. Header information includes the following:

- Subject - obtained from the user (optional) but may be added automatically by default when certain templates are used.

- Abstract - obtained from the user (optional).

- Keywords - selected from a list provided by a keyword application (optional).

- Author - automatically generated and required.

- Date - automatically generated and required.

Finally any 'SEE ALSO' links (see section 3.1.7) established by the author will be resolved at this time.

3.1.6 Implied publication

There are certain instances where publication will be implied. Examples include automatic change notifications from DIRCCE and automatic 'SEE ALSO' link generation from DIRCCE.

3.1.7 Link creation

The author of a MECE entry may establish two types of links, "reference" and "see-also", between the current document and an existing document. Links are hypertext-style, allowing for easy browsing of authored entries. Reference links are established from the current document to an existing document. In later versions of MECE, users will be able to link to a specific point in an existing document. See-also links point from an existing document to the current document. Links may exist between notebooks, both private and public. While a user may establish a link from his private notebook to a public notebook, links in the reverse direction will not be supported.

3.1.8 Automatic notification

Automatic notification of interested users, that there are new entries to a notebook, will be handled at publication time. Users will be notified of all new entries that match a set of criteria specified by them.

3.2 Navigation

Notebook entries may be found by searching or by following hypertext reference or see-also links.

3.2.1 Finding entries by searching

Searching for an entry may be accomplished in several ways. First, the entry may be selected from a graphical browser showing the links between entries. The graphical browser display may be controlled by specifying a starting entry, direction, and depth.

Alternatively, entries may be retrieved via a controlled search. Possible search parameters are the following:

- Author - a list of subsets of known authors.
- Date - a list of ranges of dates.
- Abstract - a regular expression text search.
- Subject - a regular expression text search.
- Keyword - a list of keywords chosen from the keyword application.
- Links - starting entries, direction, and depth are specified.

Additional search modes may be provided. These include searching for the most recent entries, and developing search perscriptions by combining search control parameters using "and" and "or" operations.

3.3 Mail

In essence, the top-level mail capability is equivalent to navigation applied to the mail spool rather than the project notebook. In this case, the mail spool is regarded simply as another notebook. Both creating and replying to MECE MM mail messages will be accomplished using the authoring tool.

3.4 Applications

There are essentially three types of applications in the MECE system. These types are fully external, integrated, and keyword. Both the fully external and integrated groups may contain one or more applications. The keyword type really consists of only a single general application with several uses.

3.4.1 Fully external applications

This group consists of applications that are external and that can be launched from within the MECE system. The results from the execution of an external application may be captured via a snapshot, an imported bit map or animation, or recorded video (screen grab) and used to author a new MECE MM entry. An example of an external application is the I-DEAS modeling package where data and results are not in the form of MECE MM elements. Therefore, I-DEAS results might be included in a MECE entry by taking a screen snapshot of the results and by pasting it into the MECE entry.

3.4.2 Integrated applications

Integrated applications are those whose results are presented in the form of MECE MM elements so that thesy can be edited directly with an authoring tool. Examples of integrated applications are DIRCCE and the Engineer's Scratch Pad (ESP). DIRCCE offers a form of implicit authoring via automatic notification of updates since a MECE entry may contain a reference to a drawing file which may be subsequently updated. ESP can also be modified so that MECE can manipulate the plots it generates.

3.4.3 Keyword (taxonomy) application

The taxonomy,or keyword application is a general MECE application which may be used mainly for managing keywords. Other uses may be for a product browser or for an active organization chart. Essentially, the keyword application will generate sets of keywords that form a tree. Each keyword will be defined by specifying the following attributes: definition, synonyms, parent and children, and perhaps an author. The intersection of the synonyms of distinct keywords must be empty, and no keyword may be the synonym of another. Several operations may be performed on the keyword tree. These include browsing and selecting a keyword, listing the synonyms for a keyword, performing a text search, finding all descendants or ancestors of a keyword, and finding the descendants or ancestors **n** levels removed.

3.5 Help

MECE will provide a system level hypertext interface to a help facility. The MECE user will be able to browse all the available documentation on the functionality of the MECE system via the hypertext help interface at the top level. In addition, MECE will also contain context-specific help for each significant MECE activity and command.

3.6 Conferencing And Shared Applications

Conferencing will provide the capability to simulate a group of people clustered around a computer. Functionality will include audio, video, shared applications, and multiple workspaces. MECE users will have the ability to capture real time data via a circular buffer storing audio, recorded external video, and recorded simulated video of the screen. A snapshot of the conference may be forced at any time and used to author a MECE entry. Any segment of the circular buffer can also be copied and pasted into an authoring tool. The buffer is circular in that only the last **n** minutes of the conference are stored and can be retrieved.

ACRONYMS, NAMES, AND ABBREVIATIONS

ABSML	A Better Structured Markup Language(MECE extension to HTML)
AI	Artificial Intelligence
ARPA	Advanced Research Projects Agency
CAD	Computer Aided Design
CE	Concurrent Engineering
DARPA	Defense Advanced Research Projects Agency
DICE	DARPA Initiative in Concurrent Engineering
DIRCCE	Distributed Revision Control for Concurrent Engineering(Research software, Lockheed Artificial Intelligence Center)
EDN	Electronic Design Notebook
FEM	Finite Element Method
FTP	File Transfer Protocol
HTML	HyperText Markup Language, based upon SGML
HTTP	HyperText Transport Protocol
IPC	Interprocess Communication
IRAD	Independent Research And Development
LAN	Local Area Network
MECE	Multimedia Engineering Collaborative Environment
MFC	Microsoft Foundation Class Library(PC interface development software)
MIME	Multipurpose Internet Mail Extensions
MM	MultiMedia
Motif	UNIX interface development software
MPEG	Motion Picture Experts Group
Q F D	Quality Function Deployment
R C S	Revision Control System(Public domain software)
SBD	Simulation-based Design

SHADE	Shared Dependency Engineering
TCP/IP	Transmission Control Protocol/Internet Protocol(Public domain networking protocols)
UNIX	Operating system software(Commercial software, AT&T Bell Laboratories)
VNS	Virtual Notebook System(Commercial EDN software, The ForeFront Group, Inc., Houston, Texas)
VRML	Virtual Reality Markup Language(3D model specification language)
WAIS	Wide-Area Information Servers
WAN	Wide Area Network(such as the Internet)
WWW	World Wide Web